Asian Journal of Information Technology 15 (16): 2988-2991, 2016
ISSN: 1682-3915
© Medwell Journals, 2016

Burr Type X Software Reliability Growth Model

[1]Gutta Sridevi and [2]Shaik Akbar
[1]Department of Computer Science and Engineering, KL University, Guntur, India
[2]Department of Computer Science and Engineering, Andhra Loyola Institute of Engineering and Technology, Vijayawada, India

Abstract: Software industry is on rise nowadays and it has become the part and parcel in the lives of human beings. Such software when used as an application should give the customer a feeling of trustworthiness which is nothing but technically termed as reliability. The system should be able to perform the required functions according to the stated conditions for a specific period of time. It is very important to maintain the reliability of the software to keep track on the important information which includes resources, capital, details of employees, etc., software plays an equal role along with hardware and therefore the evaluation and measurement are done equally. The most challenging task for the industry is to develop reliable software which consumes more amount of time and expensive too. This study was performed with the purpose of estimating the reliability based on Burr type X software reliability growth model. In this case we adopt a testing called as reliability testing whose purpose is to discover the potential problems in the phases of development like design as soon as possible and ultimately provide the belief that the system meets its reliability requirements.

Key words: Maximum likelihood estimation, non homogeneous poisson process, burr type X distribution and time domain data, reliability, design

INTRODUCTION

Reliability is the system's ability to perform required functioning under stated conditions for some specified time period (Lyu, 1996; Musa *et al*., 1987). Various parameters can improve the software reliability. It's important to maintain the software reliability to keep track of information details of transactions, employees, money etc., nowadays there is a tremendous increase in the size and complexity of the systems Due to which it's becoming very difficult to maintain the software reliability. Reliability is related to systems safety and we use some common methods for its analysis. Reliability focuses on the failures caused by various threats. To improve the software reliability various approaches can be used. It's very difficult to maintain the time of development and budget with reliability of the software. The best way to attain software reliability to develop high quality software that goes through all the stages of software life cycle. We should identify or recognize the failure rate between hardware and software. When the component is manufactured initially there will more faults but later on they will be minimized when the faults are recognized and rectified. With the help of useful life phases few faults can be identified.

The fault rate increases when the component physically wears out. Its not always possible to test all the requirements of the system. Some systems are very expensive to test and may take years and years to observe the faults. Some tests will require limited use of the resources or test ranges. In that case some different approaches for the testing can be made. Gompertz reliability can be used to analyze the success or the failure of the software. For software the error rate is higher at level of integration testing. These errors can be identified and removed at the slower rate during its operational use. To the producer and consumer different levels of test plans can result in different risks. Software fault prediction is one of the most important quality assurance activities in the software quality engineering stream. Many of the software systems like those implemented in telecommunication and medical areas require a very high level of quality assurance. The management of these systems necessitates an evaluating process of the quality of software modules which can be done by implementing software fault prediction techniques. A project manager or members of quality assurance group can improve the product quality by assigning necessary budget and human resources to deal with the fault-prone modules identified by fault prediction models i.e., software reliability growth models.

Literature review: Burr (1942) introduced twelve different forms of cumulative distribution functions for modeling data. We consider the three-parameter Burr-type X distribution (Prasad *et al*., 2014). It is typically defined in terms of its cumulative distribution function CDF:

$$F(x;\alpha,\lambda) = \left(1 - e^{-(\lambda x)^2}\right)^{\alpha}; x > 0, \alpha > 0, \lambda > 0 \quad (1)$$

where, α and λ are shape and scale parameters respectively. To detect the faults in the software we have found two sets of software failure data: Time domain data and Interval domain data. The time domain data is characterized by recording the individual times at which the failure occurred. The interval domain data is characterized by counting the number of failures occurring during a fixed period.

This study proposes Burr type X primarily based code reliability growth model with time domain data. So here we are given with an equation where we estimate the unknown parameters. The unknown parameters of the model are estimated using the Maximum Likelihood (ML) estimation technique. Reliability of a software system using Burr type X distribution that relies on Non Homogenous Poisson Process (NHPP) is conferred through estimation procedures. In probability theory, non-homogeneous Poisson process is a Poisson process with rate unknown parameter such that the rate parameter of the process is a function of time. The performance of the SRGM is judged by its ability to suit the software failure knowledge. However smart will a mathematical model suitable the data is additionally being calculated. To access the performance of the thought of SRGM, we've applied the parameter estimation on the real software failure datasets.

NHPP model: The Non homogeneous poisson process based software reliability growth model for N-version programming systems based on the non homogeneous Poisson process. Due to continuous extraction of errors from the software versions the people don't consider growth of reliability in N-version programming systems. A debugging effort is used to remove the errors or faults during the debugging and testing the product. New errors and faults may be introduced instead of removing them successfully due the complexity of the system. The new N-VP software reliability growth model is established in which the multi version failures which are coincident modeled by applying the generalized Non Homogeneous Poisson Process (NHPP) model into N-Version Programming (NVP) system (Teng and Pham, 2002). The application of this new software reliability can be illustrated by estimation of system reliability they are provided with the s-confidence bound.

To predict the NVP systems performance and for the evaluation of the reliability this model can be used. The N-VP SRGM is also used for overcoming the short comes of reliability model which is independent. The systems reliability can be predicted much accurately than any model which is independent and can also be used to determine when the testing should be stopped. It is a key question in N-VP SDLC.

There are independent and common faults. Faults cannot be removed successfully and new faults are introduced into the system while debugging process. The system reliability cannot grow if the faults are introduced in the debugging process which cannot be done in function of intensity failure. In NVP system different faults in that system has different roles. If the faults are common for multiple versions then they will be activated with the same input then multiple versions will fail. Suppose N (t) is known to have a Poisson probability mass function with parameters m (t), i.e.:

$$P\left[N(t) = y\right] = \frac{e^{-m(t)}\left[m(t)\right]^y}{y!}, \quad (2)$$
$$y = 0, 1, 2, \ldots$$

Then, N (t) is called an NHPP. The conclusion reached on the basis of the paper are how the data of a particular model is being tested and which standard method it is following. Various types of reliability growth models are being tested. The software reliability growth model assumes testing can be performed randomly or homogeneously. With the help of some mechanism which is random the data of test can be chosen from the input and the testing of the software can be done assuming the conditions which are homogeneous. The number of test runs will be conducted during the phase of testing. The input is given in such a way that testing will be effective and we can recognize more faults. If any failure has occurred then many test runs will be conducted to avoid it or to reduce it. Clusters occur when failures have a chance to occur in group which is also known as cluster.

The system fails when automatically two versions of the software fail. The mechanism of decision is perfect as assumed and there are two categories as discussed above that is common modes of failure and S-independent modes. Due to software faults the system will fail while execution. The failures of the software follow NHPP. Many SRGM which are dependent on NHPP became more successful tools in SRE, i.e., software reliability engineering.

The failure of the software rate is directly proportional to the no. of remaining faults in the software by that time. Due to software failure the effort of debugging occurs immediately which will remove the corresponding faults.

MATERIALS AND METHODS

Maximum likelihood estimation: One of the statistical models used for estimating the parameters is known as Maximum Likelihood Estimation (MLE).

In statistics it is a well-known estimation method. For example if we want to find the heights of adult Ostrich and it is difficult for us to measure each and every individual heights from every nook and corner of the world due to some constraints like time, cost, etc. So, we go for normal distribution assuming that the heights are normally distributed with mean and variance which are unknown. The values of mean and variance can be estimated by the method of MLE by knowing the heights of few ostrich which means that a sample form all the population is taken and is estimated. Ronald Fisher proposed this theory in between 1912 and 1922. Some failure data sets are taken and using the derivation mentioned above we have estimated the parameter values and calculated reliability for each dataset:

$$f\left(x_i \middle| n, w\right) = \frac{n!}{(n-x_i)!x_i!} \left(w_1 \exp\right)\left(-w_2 t_i\right))^{xi} \left(1 - w_1 \exp\left(-w_2 t_i\right)\right)^{n-xi} \tag{3}$$

The mean value function of Burr type X model is given by:

$$m(t) = a\left[1 - e^{-(tc)^2}\right]^b \tag{4}$$

The Probability Density Function (PDF) of Burr XII distribution is given by:

$$\lambda(t) = 2abc^2 t.e^{-(tc)^2}\left[1 - e^{-(tc)^2}\right]^{b-1} \tag{5}$$

The likelihood function for time domain data is given by:

$$\text{LogL} = \sum_{i=1}^{n} \log\left(\lambda(t_i)\right) - m(t_n) \tag{6}$$

$$\text{LogL} = \sum_{i=1}^{n} \text{Log}\left[2abc^2 t_i.e^{-(t_i c)^2}\left\{1 - e^{-(t_i c)^2}\right\}^{b-1}\right] - a\left[1 - e^{-(t_n c)^2}\right]^b \tag{7}$$

Taking partial derivative with respect to 'a' and equating it to 0:

$$\frac{\partial \text{LogL}}{\partial a} = \sum_{i=1}^{n} \frac{1}{a} - \left[1 - e^{-(t_n c)^2}\right]^b \tag{8}$$

$$\therefore\ a = \frac{n}{\left[1 - e^{-(t_n c)^2}\right]^b} \tag{9}$$

Parameter 'b' is estimated using Newton raphson's Method, we get:

$$\text{Log L} = \sum_{i=1}^{n} \left[\begin{array}{l} \log 2 + \log a + \log b + 2\log c + \log t_i + \\ \log\left(e^{-(t_i c)^2}\right) + \log\left\{1 - e^{-(t_i c)^2}\right\}^{b-1} \end{array}\right] - a\left[1 - e^{-(t_n c)^2}\right]^b \tag{10}$$

$$\frac{\partial \text{LogL}}{\partial b} = \frac{n}{b} + \sum_{i=1}^{n} \log\left[1 - e^{-(t_i c)^2}\right] - n.\log\left[1 - e^{-(t_n c)^2}\right] \tag{11}$$

$$\frac{\partial^2 \text{LogL}}{\partial b^2} = \frac{-n}{b^2} \tag{12}$$

Parameter 'c' is also estimated using Newton Raphson Method:

$$\frac{\partial \text{LogL}}{\partial c} = \frac{2n}{c} - 2t_i^2 c - \frac{2ct_i^2(b-1)(e^{-(t_i c)^2})}{1 - e^{-(t_i c)^2}} - \frac{2nt_n c^2 b e^{-(t_n c)^2}}{\left[1 - e^{-(t_n c)^2}\right]} \tag{13}$$

$$\frac{\partial^2 \text{LogL}}{\partial c} = \sum_{i=1}^{n} (t_i)^2 - \frac{2n}{c^2} - 2t_n - \frac{2nt_n^2 e^{-(t_n c)^2}}{1 - e^{-(t_n c)}} + \frac{2c^2 t_n^2 e^{-(t_n c)^2}}{\left(1 - e^{-(t_n c)}\right)^2} \tag{14}$$

RESULTS AND DISCUSSIONS

The set of software failure data analyzed here is borrowed from software development project as published in Pham (2003, 2006) and Asoka (2010).

CONCLUSION

The conclusions reached on the basis of the study are how the data of a particular model is being tested

Table 1: Estimation of parameters

Datasets	Samples	a	b	c
IBM	15	15.00394	0.022493	0.007132
NTDS	26	26.00627	0.028927	0.008755
Xie	30	30.0024	0.011503	0.003015
SONATA	30	30.00591	0.003318	0.000912
AT and T	22	22.00148	0.011863	0.003346

Table 2: Datasets and its reliabilities

Datasets	Reliability
IBM	0.99683
NTDS	0.99449
Xie	0.99878
SONATA	0.99908
AT and T	0.99919

and which standard method it is following. Various types of reliability growth models are being tested. The software reliability growth model assumes testing can be performed randomly or homogeneously. With the help of some mechanism which is random the data of test can be chosen from the input and the testing of the software can be done assuming the conditions which are homogeneous. The number of test runs will be conducted during the phase of testing. The input is given in such a way that testing will be effective and we can recognize more faults. If any failure has occurred then many test runs will be conducted to avoid it or to reduce it. The above approach is validated by estimating the parameters by using different datasets.

The experiment result shows that, the AT and T dataset got the maximum reliability value therefore AT and T dataset is more reliable for Burr type X which is shown in Table 1 and 2.

REFERENCES

Burr, I.W., 1942. Cumulative frequency functions. Ann. Math. Stat., 13: 215-232.

Lyu, M.R., 1996. Handbook of Software Reliability Engineering. McGraw-Hill and IEEE Computer Society, New York, ISBN: 0-07-039400-8.

Musa, J.D., A. Iannino and K. Okumoto, 1987. Software Reliability: Measurement Prediction Application. McGraw Hill, New York, USA., ISBN:9780070440937, Pages: 621.

Pham, H., 2003. Handbook of Reliability Engineering. 1st Edn., Springer-Verlag, London Berlin Heidelberg, ISBN 1-85233-453-3.

Pham, H., 2006. System Software Reliability. Springer, Berlin, Germany, ISBN-13: 978-1-85233-950-0, Pages: 440.

Prasad, R.S., K.M. Mohan and G. Sridevi, 2014. Burr type XII software reliability growth model. Int. J. Comput. Appl., 108: 16-16.

Teng, X. and H. Pham, 2002. A software-reliability growth model for N-version programming systems. IEEE. Trans. Reliab., 51: 311-321.